

Program burnset.c

```
/* This software was developed at the National Institute of */
/* Standards and Technology by employees of the Federal Government */
/* in the course of their official duties. Pursuant to title 17 */
/* Section 105 of the United States Code this software is not */
/* subject to copyright protection and is in the public domain. */
/* CEMHYD3D is an experimental system. NIST assumes no */
/* responsibility whatsoever for its use by other parties, and */
/* makes no guarantees, expressed or implied, about its quality, */
/* reliability, or any other characteristic. We would appreciate */
/* acknowledgement if the software is used. This software can be */
/* redistributed and/or modified freely provided that any */
/* derivative works bear some notice that they are derived from it, */
/* and any modified versions bear some notice that they have been */
/* modified. */

#define BURNT 70      /* label for burnt pixels */
#define SIZESET 100000
/* Transformation functions for changing direction of burn propagation
*/
#define cx(x,y,z,a,b,c) (1-b-c)*x+(1-a-c)*y+(1-a-b)*z
#define cy(x,y,z,a,b,c) (1-a-b)*x+(1-b-c)*y+(1-a-c)*z
#define cz(x,y,z,a,b,c) (1-a-c)*x+(1-a-b)*y+(1-b-c)*z

/* routine to assess connectivity (percolation) of solids for set
estimation */
/* Definition of set is a through pathway of cement and fly ash (slag)
*/
/* particles connected together by CSH, C3AH6, or ettringite */
/* Two matrices are used here: one to store the recently burnt
locations */
/*           the other to store the newly found burnt locations */
int burnset(d1,d2,d3)
    int d1,d2,d3;
{
    long int ntop,nthrough,icur,inew,ncur,nnew,ntot,count_solid;
    int i,j,k,seyet;
    static int nmatx[SIZESET],nmaty[SIZESET],nmatz[SIZESET];
    int xl,xh,j1,k1,px,py,pz,qx,qy,qz;
    int xcn,ycn,zcn,x1,y1,z1,igood;
    static int nnewx[SIZESET],nnewy[SIZESET],nnewz[SIZESET];
    int jnew;
    float mass_burn=0.0,alpha_burn=0.0,con_frac;
    FILE *percfile;
    static char newmat [SYSIZE] [SYSIZE] [SYSIZE];

/* counters for number of pixels of phase accessible from surface #1 */
/* and number which are part of a percolated pathway to surface #2 */
    ntop=0;
    nthrough=0;
    seyet=0;
    for(k=0;k<SYSIZE;k++){
        for(j=0;j<SYSIZE;j++){
            for(i=0;i<SYSIZE;i++){
```

```

        newmat[i][j][k]=mic[i][j][k];

    }
}
}

/* percolation is assessed from top to bottom only */
/* in transformed coordinates */
/* and burning algorithm is periodic in other two directions */
i=0;

for(k=0;k<SYSIZE;k++){
for(j=0;j<SYSIZE;j++){

    igood=0;
    ncur=0;
    ntot=0;
    /* Transform coordinates */
    px=cx(i,j,k,d1,d2,d3);
    py=cy(i,j,k,d1,d2,d3);
    pz=cz(i,j,k,d1,d2,d3);

/* start from a cement clinker, slag, fly ash ettringite, C3AH6, or CSH
pixel */
    if((mic [px] [py] [pz]==C3S) ||
       (mic [px] [py] [pz]==C2S) ||
       (mic [px] [py] [pz]==SLAG) ||
       (mic [px] [py] [pz]==ASG) ||
       (mic [px] [py] [pz]==CAS2) ||
       (mic [px] [py] [pz]==POZZ) ||
       (mic [px] [py] [pz]==CSH) ||
       (mic [px] [py] [pz]==C3AH6) ||
       (mic [px] [py] [pz]==ETTR) ||
       (mic [px] [py] [pz]==ETTRC4AF) ||
       (mic [px] [py] [pz]==C3A) ||
       (mic [px] [py] [pz]==C4AF)){
           /* Start a burn front */
           mic [px] [py] [pz]=BURNT;
           ntot+=1;
           ncur+=1;
           /* burn front is stored in matrices nmat* */
           /* and nnew* */
           nmatx[ncur]=i;
           nmaty[ncur]=j;
           nmatz[ncur]=k;
           /* Burn as long as new (fuel) pixels are found
*/
       do{
           nnew=0;
           for(inew=1;inew<=ncur;inew++){
               xcn=nmatx[inew];
               ycn=nmaty[inew];
               zcn=nmatz[inew];
               /* Convert to directional
coordinates */
               qx=cx(xcn,ycn,zcn,d1,d2,d3);
               qy=cy(xcn,ycn,zcn,d1,d2,d3);
               qz=cz(xcn,ycn,zcn,d1,d2,d3);

```

```

        /* Check all six neighbors */
        for(jnew=1;jnew<=6;jnew++){
            x1=xcn;
            y1=ycn;
            z1=zcn;
            if(jnew==1){x1-=1;}
            if(jnew==2){x1+=1;}
            if(jnew==3){y1-=1;}
            if(jnew==4){y1+=1;}
            if(jnew==5){z1-=1;}
            if(jnew==6){z1+=1;}
            /* Periodic in y and */
            if(y1>=SYSIZE){y1-
                =SYSIZE;}
            else
                /* Periodic in z
                direction */
                if(z1>=SYSIZE){z1-
                =SYSIZE;}
            else
                if(z1<0){z1+=SYSIZE;}
            /* Nonperiodic so be sure to remain in the 3-D box */
            if((x1>=0)&&(x1<SYSIZE)){
                px=cx(x1,y1,z1,d1,d2,d3);
                py=cy(x1,y1,z1,d1,d2,d3);
                pz=cz(x1,y1,z1,d1,d2,d3);
                /* Conditions for propagation of burning */
                /* 1) new pixel is CSH or ETTR or C3AH6 */

                if((mic[px][py][pz]==CSH)|| (mic[px][py][pz]==ETTRC4AF)|| (mic[px][py][pz]
                ==C3AH6)|| (mic[px][py][pz]==ETTR)){
                    ntot+=1;
                    mic [px] [py] [pz]=BURNT;
                    nnew+=1;
                    if(nnew>=SIZESET){
                        printf("error in size of nnew %d\n",
                        nnew);
                    }
                    nnewx[nnew]=x1;
                    nnewy[nnew]=y1;
                    nnewz[nnew]=z1;
                }
                /* 2) old pixel is CSH or ETTR or C3AH6 and new pixel is one of cement
                clinker, slag, of fly ash phases */
                else
                if(((newmat[qx][qy][qz]==CSH)|| (newmat[qx][qy][qz]==ETTRC4AF)|| (newmat[
                qx][qy][qz]==C3AH6)|| (newmat[qx][qy][qz]==ETTR))
                    &&((mic [px] [py] [pz]==C3S) ||
                    (mic [px] [py] [pz]==C2S) ||
                    (mic [px] [py] [pz]==CAS2) ||

```

```

(mic [px] [py] [pz]==SLAG) ||
(mic [px] [py] [pz]==POZZ) ||
(mic [px] [py] [pz]==ASG) ||
(mic [px] [py] [pz]==C3A) ||
(mic [px] [py] [pz]==C4AF))){ ntot+=1;
mic [px] [py] [pz]=BURNT;
nnew+=1;
if(nnew>=SIZESET){
printf("error in size of nnew %d\n",
nnew);
}
nnewx[nnew]=x1;
nnewy[nnew]=y1;
nnewz[nnew]=z1;
}
/* 3) old and new pixels belong to one of cement clinker, slag,
or fly ash phases and */
/* are contained in the same initial cement particle */
/* and it is not a one-pixel particle */
else if((micpart[qx][qy][qz]==micpart[px][py][pz])
&&(micpart[qx][qy][qz]!=0)
&&((mic [px] [py] [pz]==C3S) ||
(mic [px] [py] [pz]==C2S) ||
(mic [px] [py] [pz]==POZZ) ||
(mic [px] [py] [pz]==SLAG) ||
(mic [px] [py] [pz]==ASG) ||
(mic [px] [py] [pz]==CAS2) ||
(mic [px] [py] [pz]==C3A) ||
(mic [px] [py]
[pz]==C4AF ))&&((newmat[qx][qy][qz]==C3S) ||
(newmat [qx] [qy] [qz]==C2S) ||
(newmat [qx] [qy] [qz]==SLAG) ||
(newmat [qx] [qy] [qz]==ASG) ||
(newmat [qx] [qy] [qz]==POZZ) ||
(newmat [qx] [qy] [qz]==CAS2) ||
(newmat [qx] [qy] [qz]==C3A) ||
(newmat[qx][qy][qz]==C4AF))){ ntot+=1;
mic [px] [py] [pz]=BURNT;
nnew+=1;
if(nnew>=SIZESET){
printf("error in size of nnew
%d\n", nnew);
}
nnewx[nnew]=x1;
nnewy[nnew]=y1;
nnewz[nnew]=z1;
}
} /* nonperiodic if delimiter */
*/
} /* neighbors loop */
} /* propagators loop */
if(nnew>0){
ncur=nnew;
/* update the burn front
matrices */

```

```

        for(icur=1;icur<=ncur;icur++){

nmatx[icur]=nnewx[icur];

nmaty[icur]=nnewy[icur];

nmatz[icur]=nnewz[icur];
}
}
}while (nnew>0);

ntop+=ntot;
x1=0;
xh=SYSIZE-1;
/* Check for percolated path through system */
for(j1=0;j1<SYSIZE;j1++){
for(k1=0;k1<SYSIZE;k1++){
    px=cx(xl,j1,k1,d1,d2,d3);
    py=cy(xl,j1,k1,d1,d2,d3);
    pz=cz(xl,j1,k1,d1,d2,d3);
    qx=cx(xh,j1,k1,d1,d2,d3);
    qy=cy(xh,j1,k1,d1,d2,d3);
    qz=cz(xh,j1,k1,d1,d2,d3);
    if((mic [px] [py] [pz]==BURNT)&&(mic [qx] [qy]
[qz]==BURNT)){
        igood=2;
    }
    if(mic [px] [py] [pz]==BURNT){
        mic [px] [py] [pz]=BURNT+1;
    }
    if(mic [qx] [qy] [qz]==BURNT){
        mic [qx] [qy] [qz]=BURNT+1;
    }
}
}

if(igood==2){
    nthrough+=ntot;
}
}
}
}

printf("Phase ID= Solid Phases \n");
printf("Number accessible from first surface = %ld
\n",ntop);
printf("Number contained in through pathways= %ld
\n",nthrough);
percfile=fopen(ptsname,"a");
mass_burn+=specgrav[C3S]*count[C3S];
mass_burn+=specgrav[C2S]*count[C2S];
mass_burn+=specgrav[C3A]*count[C3A];
mass_burn+=specgrav[C4AF]*count[C4AF];
alpha_burn=1.-(mass_burn/cemmass);
con_frac=0.0;

```

```

        count_solid=count[C3S]+count[C2S]+count[C3A]+count[C4AF]+count[ET
TR]+count[CSH]+count[C3AH6]+count[ETTRC4AF]+count[POZZ]+count[ASG]+coun
t[SLAG]+count[CAS2];
        if(count_solid>0){
            con_frac=(float)nthrough/(float)count_solid;
        }
        fprintf(percfile,"%ld %f %f %ld %d
%d\n",cycCnt,time_cur+(2.*(float)(cycCnt)-
1.0)*beta/krate,alpha_burn,nthrough,count[C3S]+count[C2S]+count[C3A]+co
unt[C4AF]+count[CAS2]+count[SLAG]+count[ASG]+count[POZZ]+count[ETTR]+co
unt[C3AH6]+count[ETTRC4AF]+count[CSH],con_frac);
        fclose(percfile);
        if(con_frac>0.985){setyet=1;}

/* return the burnt sites to their original phase values */
    for(i=0;i<SYSIZE;i++){
        for(j=0;j<SYSIZE;j++){
            for(k=0;k<SYSIZE;k++){
                if(mic [i] [j] [k]>=BURNT){
                    mic [i] [j] [k]= newmat [i] [j] [k];
                }
            }
        }
    }
/* Return flag indicating if set has indeed occurred */
return(setyet);
}

```